Deux Etapes pour la Création Dynamique d'Activités Pédagogiques

Claude MOULIN**, **Nadia CATENAZZI***, **Lorenzo SOMMARUGA***

**CRS4, Sesta Strada Ovest, Z.I. Macchiareddu, Uta (CA), 09010, Italie
{claumou@ciaoweb.it}
* MEDIATECH S.R.L., C.P. 100, 09018 Sarroch (CA), ITALIE
{nadia.catenazzi@bigfoot.com, l.sommaruga@fantastic.com}

Résumé. La recherche sur les plate-formes d'apprentissage à distance vise à définir des modèles pour la structure, le déploiement, la gestion et l'utilisation de ces environnements intelligents. Dans ce contexte, cet article présente un modèle de génération dynamique d'activités pédagogiques. L'idée de base est que les activités pédagogiques ne sont ni statiques ni prédéfinies, mais qu'elles peuvent être créées dynamiquement à partir de documents de base et présentées de différentes façons selon le modèle de l'étudiant. Les documents de base ne sont pas associés à une seule activité, mais au contraire sont conçus pour être réutilisés dans différentes activités.
L'article présente en particulier une implémentation d'une activité de résolution de problème, générée par un double processus, et qui démontre comment l'adaptivité et la réutilisabilité peuvent être réellement effectives grâce à l'intégration de technologies telles que XML, DOM, XSL, et la puissance du langage Java au travers des servlets et de Jsp.
Mots-clés. Environnements intelligents, Apprentissage à distance, Activités pédagogiques, Documents virtuels, Paradigme XML, Adaptivité dynamique.

Abstract. *Research on distance learning platforms aims to define models for the structure, deployment, management and use of these intelligent environments. Within this context, this paper presents a model of dynamic generation of pedagogical activities. The basic idea is that pedagogical activities are neither static nor predefined, but they can be dynamically created starting from basic documents, and presented in different ways according to the student model. The basic documents are not associated to only one activity. They are conceived to be re-used in different activities.*
The paper presents in particular an implementation of a problem solving activity, generated from a two-step process, which shows how adaptability and reusability can be really effective thanks to the integration of tecnologies such as XML, DOM, XSL and the potential of the Java language by using servlets and Jsp.
Key-words. *Intelligent environments, Distance learning, Pedagogical activities, Virtual documents, XML paradigm, Dynamic adaptivity.*

1. Introduction

Depuis septembre 1998, nous développons une plate-forme de recherche multi-media visant à la modélisation, la génération, la gestion et l'utilisation d'environnements d'apprentissage à distance intelligents. Les modèles que nous proposons sont centrés sur les compétences que l'environnement vise à faire acquérir et les fonctionnalités requises pour ceci.
Nous considérons que les modèles et les structures pour l'information, les connaissances et les compétences sont primordiales. Notre solution repose sur une organisation des cours et du matériel d'apprentissage basée sur une typologie d'activités spécifiques proposées aux étudiants. Les contenus pédagogiques utilisés doivent être au service des activités proposées à un utilisateur et non l'inverse. D'un point de vue technique, ceci est rendu possible maintenant par la technologie XML associée à la technologie Java au travers des applications Web [9].
Nous soutenons avec force l'idée de réutilisabilité de méthodes et de ressources. Il faut pour ceci obtenir un consensus assez large au sein d'une même communauté. Dans cet article, nous insistons sur le modèle d'activités que nous proposons et la façon de les présenter dynamiquement à partir de contenu de base généraux. Ce modèle est à la fois ouvert pour laisser place à la créativité des auteurs et contraint à un certain endroit pour assurer la réutilisabilité.
Cet article donne initialement une vue d'ensemble d'un environnement d'apprentissage à distance en insistant sur la modélisation des activités pédagogiques. La deuxième partie montre les deux processus qui exécutés en chaîne, permettent de présenter dynamiquement une activité. La troisième partie est dédiée à l'implémentation des processus au travers de la description d'un exemple concret dans le domaine de la géométrie euclidienne.

2. Description d'un environnement d'apprentissage à distance

L'architecture d'une plate-forme d'apprentissage que nous préconisons est composée de différents modules correspondant aux acteurs et aux fonctionalités impliqués dans le processus d'apprentissage [3]. Chaque interaction dans ce processus réfère aux mêmes éléments qui forment les structures de base du matériel pédagogique et l'infrastructure du processus d'apprentissage. Les auteurs et les étudiants utilisent ces mêmes structures, mais d'une façon différente. Comme principales structures nous ne décrirons ici que les compétences et les connaissances qui s'y rattachent, les cours, les activités pédagogiques, les contenus et le modèle de l'étudiant.
L'objectif principal d'une plate-forme d'apprentissage est de faire acquérir des compétences. Celles-ci peuvent être classées par exemple, selon un domaine particulier (mathématiques, langues, informatique, …) et la taxonomie de Bloom (Connaissance, Compréhension, Application, Analyse, Synthèse et Evaluation) [1, 7]. Une compétence, exprimée à travers un objectif d'apprentissage, est reliée à d'autres structures incluant les connaissances qui s'y réfèrent et les activités qui permettent de l'acquérir. Un cours est une liste d'objectifs d'apprentissage qui visent un but pédagogique précis. Les objectifs d'apprentissage sont les mêmes pour tous les étudiants qui suivent le même cours. La structure d'un cours contient des informations comme la liste des compétences sur lesquelles doivent se focaliser les étudiants, le public visé et les auteurs du matériel.
Les activités pédagogiques sont associées avec chaque objectif d'apprentissage qu'elles visent à faire atteindre. Au niveau supérieur, nous les classifions en deux catégories : les activités individuelles et les activités collaboratives. Des propriétés se rattachent à chaque activité spécifiant le style d'apprentissage auquel elle s'adresse, les compétences visées, un niveau de difficulté et le contenu qui la nourrit.
Les contenus représentent le matériel didactique et constituent les briques de base d'une plate-forme. Ils sont intégrés dans des documents particuliers que nous appelons *documents de base*. Ils sont utilisés pour générer les activités présentées à l'étudiant et peuvent être utilisés de différentes façons dans plusieurs activités. Les contenus doivent pouvoir être spécialement conçus par des auteurs ou importés à partir de sources externes. Ils sont alors intégrés dans le système grâce à des meta-data et des outils appropriés. L'idée innovante est de réutiliser les mêmes contenus, filtrés adéquatement, en plusieurs activités. Ceci est rendu aisément possible si les documents de base sont écrits en XML [11].
Le modèle de l'étudiant est constitué d'une partie statique, contenant des informations telles que les données personnelles et d'une partie dynamique. Celle-ci contient des informations sur les objectifs d'apprentissage initiaux, le style d'apprentissage préféré, le statut de l'apprentissage et le niveau de compétence.

3. Typologie des activités individuelles

Notre but est seulement de préciser ici les types d'activités individuelles. Citons simplement comme activités collaboratives le débat et le jigsaw [2]. Cett dernière activité consiste principalement à diviser un thème en sous-thèmes et de les attribuer chacun à un sous-groupe d'étudiants. Chaque sous-groupe doit alors étudier la partie qui lui est réservée et la présenter ensuite aux autres étudiants. A partir de travaux similaires [4, 6], nous avons classifié les activités individuelles en sept catégories qui recouvrent une bonne part des types que l'on peut trouver actuellement dans les environnements d'apprentissage :

- réponse (à un test)
- utilisation (d'une simulation)
- résolution (de problème)
- recherche (d'information),
- écriture (d'un rapport, d'un texte, d'un message),
- exploration (d'un monde),
- observation (d'un exemple, d'un élément de théorie, d'une histoire).

Pour cet article nous nous intéresserons plus particulièrement aux première et troisième catégories. Nous appelons *test* et *problème* les types de documents qui sont les supports de ces activités. Nous avons élaboré une définition de types de documents (DTD) pour chacune d'elles.

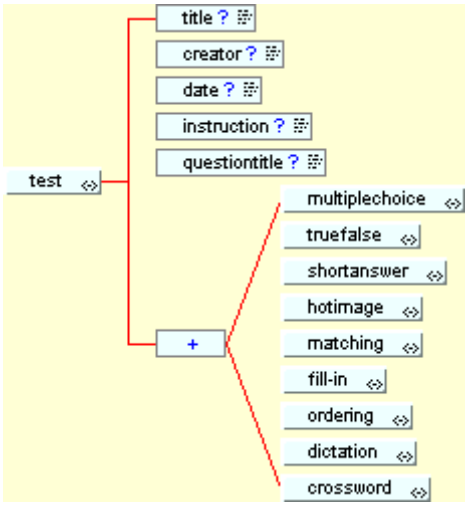

Figure 1 : DTD des *tests*

La figure 1 montre partiellement la DTD des tests. On y retrouve neuf types de questions qui peuvent s'y combiner :

- la question à choix multiple,
- le vrai-faux,
- la question à réponse courte,
- la sélection de parties d'images,
- l'association d'éléments de deux listes,
- le texte à trous,
- l'ordonnancement d'éléments d'une liste,
- la dictée
- les mots-croisés.

Les catégories ont des propriétés. Ainsi, la question à choix multiple peut accepter une ou plusieurs réponses correctes et les réponses courtes peuvent être plus ou moins dirigées. La figure 2 montre la DTD associée aux activités de résolution de problèmes. Elle est principalement composées de parties comportant une question et les solutions correspondantes.

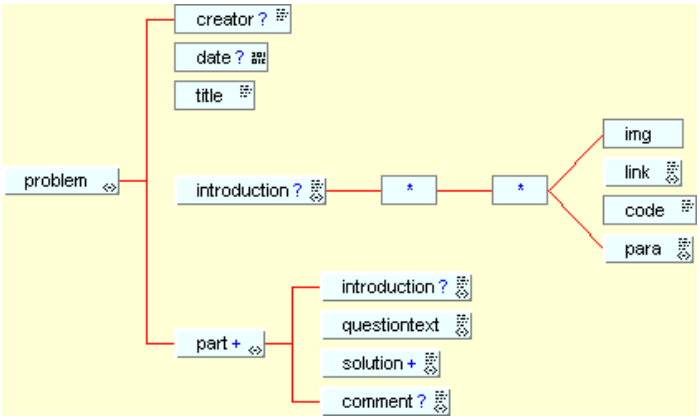

Figure 2 : DTD de la *résolution de problèmes*

Les activités sont destinées à être présentées à un étudiant dans un browser sous forme de pages écrites dans le langage HTML et élaborées dynamiquement. Pour une même activité, il existe plusieurs présentations possibles.

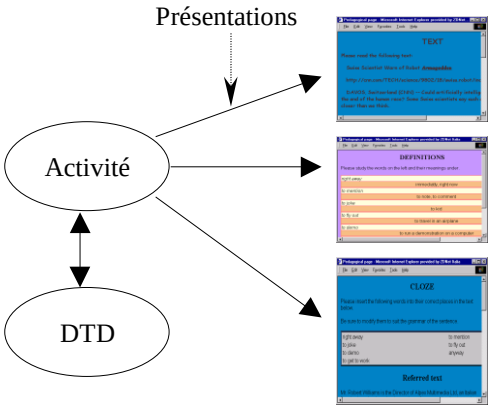

Figure 3 : présentations multiples d'une activité

Le modèle de l'étudiant que nous utilisons réfère à un style d'apprentissage et à un style tutoriel préférés. La présentation de l'activité (cf figure 3) est adaptée en utilisant ces connaissances ou, éventuellement, des connaissances de même type, que le système choisit d'utiliser en remplacement. Ce peut être le cas lors d'un cours, par exemple, pour renforcer la motivation de l'étudiant.

4. Les deux étapes de la création dynamique d'activités

Nous avons proposé dans la partie précédente une typologie d'activités qui peuvent être présentées à un étudiant. Cette typologie est décrite à l'aide de DTD. Dans l'exemple des tests que nous avons décrit, chaque partie de la DTD correspond à un type particulier d'activité. Dans le cas de la résolution de problème, la DTD n'a qu'une seule partie. Il y a ainsi une relation biunivoque entre un type de document (test ou problème) et un type d'activité proposée. Le contenu d'un document validant notre DTD est conçu pour un type unique d'activité.

Ces documents n'ont d'autre raison d'être que de servir de support à une activité. Ils ne sont destinés à autre chose. En réalité, il n'est pas envisagé dans notre système de créer directement ce type de documents pour deux raisons. Tout d'abord, leur contenu n'est pas conçu pour être réutilisé. D'autre part, les activités qui sont présentées lors d'un cours ne sont pas des entités isolées. Elles sont élément d'un ensemble d'activités. Elles doivent aider ensemble l'étudiant à acquérir des compétences.

Le niveau de documents requis pour les activités sert seulement de niveau intermédiaire. L'intérêt de ce niveau est de réduire le nombre d'outils qui doivent être créés pour rendre adaptatives les présentations d'activités. Leur nombre est réduit dans la mesure où l'ensemble des types de documents concernés est bien circonscrit.

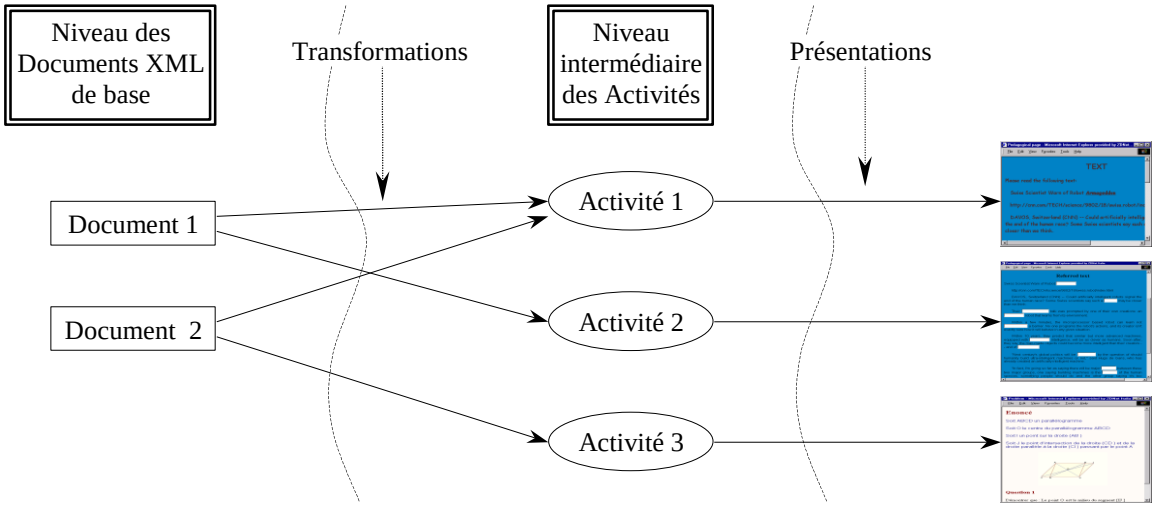

Figure 4 : documents de base et activités extraites

En revanche, nous proposons de concevoir uniquement des documents, dits documents de base [8], qui peuvent être réutilisés de différentes manières. Ainsi, un document de base doit permettre de créer plusieurs activités et une activité peut être nourrie de fragments issus de plusieurs documents de base. Nous obtenons ainsi deux niveaux de documents : le niveau des documents de base et le niveau intermédiaire des documents associés aux activités.

La première étape consiste à transformer les documents de base en documents intermédiaires standardisés, les supports d'activités, et la seconde étape consiste à présenter ces activités aux utilisateurs finaux (cf figure 4). La génération d'activités à partir des documents de base et la production du code HTML des présentations à partir des activités sont dynamiques (les procédés sont décrits plus amplement dans la partie suivante). Il suffit alors de chaîner les deux processus dynamiques pour rendre transparent le niveau intermédiaire des activités. Le code XML issu du premier processus sert d'entrée au second processus.
Les documents de base doivent donc être accompagnés des outils qui vont permettre la création des activités correspondantes. Ces outils devant produire du texte écrit dans le langage XML, il semble tout à fait naturel de créer des feuilles de style XSL [12] et de générer automatiquement grâce à un processeur comme lotusXsl [10] des fichiers XML satisfaisant les DTD d'activités que nous proposons. Les deux étapes du traitement des documents s'avèrent très pertinentes lors de l'ajout d'un nouveau type de document de base. Il suffit ensuite d'écrire les feuilles de style qui permettent de réaliser la première étape et d'obtenir ainsi les documents intermédiaires. Toute la partie présentation est alors immédiatement réutilisable.

5. Méthodes générales de présentation dynamique de documents

La construction dynamique de la présentation d'un document peut se faire de différentes façons. Le choix dépend principalement de la structure du document et du type d'interactivité désirée avec l'utilisateur. Nous avons choisi en raison du nombre d'outils disponibles d'utiliser le langage Java pour la présentation d'un document.
La façon la plus simple de présenter dynamiquement un document XML est certainement d'utiliser l'appel à une servlet qui associe une feuille de style XSL à ce document. La servlet requiert alors le service d'un processeur qui produit en réponse à partir de ces deux éléments, le document et la feuille de style, le code HTML désiré. Nous utilisons par exemple cette technique pour les activités de type *observation*.
Quelle que soit la technique utilisée, le document XML source dans tous les cas doit être lu et analysé. Le transformer en une structure d'arbre (DOM), n'est pas toujours le moyen le plus rapide ni le plus adapté pour obtenir la présentation désirée. Il est parfois plus efficace de parser le document lors de sa lecture grâce à un parseur SAX, par exemple. Créer un bean Java adapté au type de document et réalisant cette analyse est une solution intéressante. Il peut être facilement inséré dans une page Jsp et permet ainsi de déployer la puissance du code Java. Nous avons utilisé cette technique pour des activités interactives comme les *tests*.
Il est encore possible de présenter un document XML au travers d'une *applet* qui analyse elle-même le document. Encore une fois, ce peut-être une page Jsp qui à partir du type du document XML va rechercher grâce aux fichiers de configuration du site Web, l'applet désirée. Nous avons utilisé cette technique lorsque le code HTML s'avère insuffisant pour des activités fortement interactives ou pour adapter l'interactivité.

6. Implémentation d'un exemple

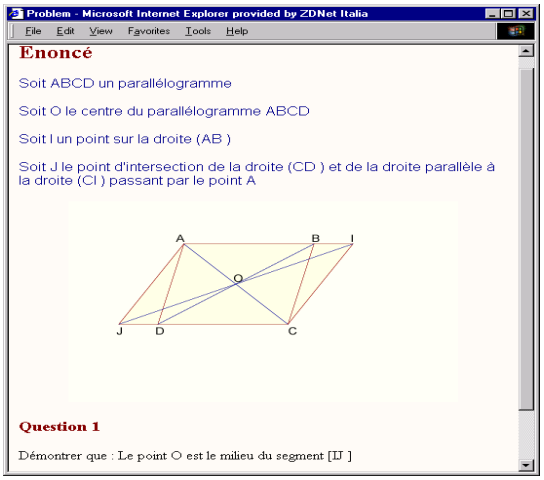

Figure 5 : Exemple de problème de géométrie

Nous présentons un exemple issu de la géométrie euclidienne pour illustrer les propos précédents. En géométrie, un exercice classique (cf. figure 5) consiste à montrer un énoncé, à construire une figure, puis à demander de démontrer qu'une propriété constatée sur la figure est vraie. Nous avons voulu écrire des fichiers XML qui permettent la construction de tels exercices mais qui soient largement réutilisables. Ce sera dans notre modèle une nouvelle sorte de documents de base.

Nous avons conçu la DTD (cf. figure 6a, 6b) pour la construction de problèmes de géométrie avec plusieurs objectifs. Premièrement, les fichiers la validant devaient pouvoir servir d'entrée à des logiciels spécifiques comme les démonstrateurs automatiques et les traceurs de figures. En second lieu, ils devaient pouvoir être présentés dans différentes langues. En troisième lieu, celui exposé ici, considérés comme documents de base, ils devaient pouvoir être transformés en activité.
Ce peut être bien évidemment l'activité de résolution de problème : demander une démonstration de la propriété géométrique. Ce peut être également un texte à trous : demander la conclusion des pas d'une démonstration (il suffit de masquer la conclusion des déductions dans les solutions). Ce peut être encore l'ordonnancement d'éléments d'une liste : demander de réordonner les étapes d'une démonstration.
Ces activités ont des niveaux de difficultés différents et sont utilisés dans différentes stratégies pédagogiques. Le processus de déduction logique est une compétence délicate et souvent difficile à acquérir et des activités de remédiation sont souvent nécessaires à employer.

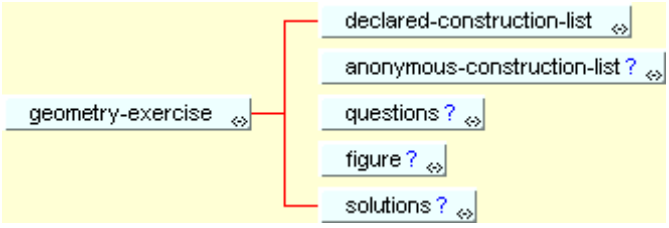

Figure 6a : Structure générale d'un problème de géométrie

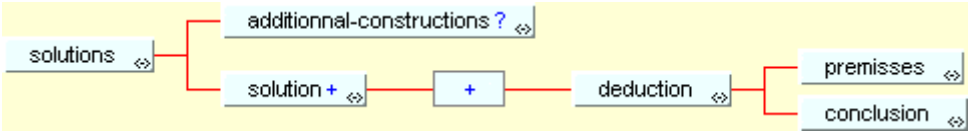

Figure 6b : Solutions d'un problème de géométrie

L'objectif n'est pas de décrire en détail la DTD. Nous donnerons seulement quelques indications générales pour une meilleure compréhension. Le fichier d'un exercice de géométrie possède tout d'abord une liste de constructions. Celles-ci permettront d'écrire des phrases de l'énoncé comme "soit ABCD un parallélogramme".
Un énoncé contient également des constructions non indiquées explicitement comme ici la construction des points A, B, C et D, une ou plusieurs questions, une figure et des solutions. Une solution est une suite ordonnée de déductions (les pas de la démonstration) comprenant chacune prémisses et conclusion. De plus chaque pas d'une solution renvoie à un théorème (nous avons créé à cet effet un document XML d'environ quatre-vingt théorèmes).
Nous insisterons quelque peu sur les constructions. Les lignes XML suivantes (cf figure 7) montrent deux constructions, celle d'un parallélogramme et celle de son centre. Chaque construction possède un identificateur qui permet de référencer l'objet construit (le built-element), cet objet lui-même et les éléments support de cette nouvelle construction. Ceux-ci doivent être préalablement construits, s'ils ne le sont déjà explicitement. Comme un élément peut servir à plusieurs constructions, il ne doit être construit qu'une seule fois et utilisé autant de fois que nécessaire grâce à son identificateur.

```
<construction constructionid="d0">
    <quadrilateral type="parallelogram" >
        <built-element type="quadrilateral" />
        <base-element constructionref="a0" role="point1" />
        <base-element constructionref="a1" role="point2" />
        <base-element constructionref="a2" role="point3" />
        <base-element constructionref="a3" role="point4" />
    </quadrilateral>
  </construction>

  <construction constructionid="d1">
      <parallelogramcentre>
          <built-element name="O" type = "point" coordinates="0, 0"/>
          <base-element constructionref = "d0"
                        role="basequadrilateral" />
      </parallelogramcentre>
  </construction>
```

Figure 7 : constructions des éléments géométriques

Partant de ce type de fichier, il est nécessaire d'écrire une feuille de style XSL qui permettra d'obtenir chacune des trois activités envisagées, la résolution de problème, le texte à trous et l'ordonnancement. Nous présentons

ci-dessous (cf figure 8) quelques lignes de la feuille de style qui permet d'obtenir le document XML support à l'activité *résolution de problème* et en particulier celle présentée dans la figure 5. La règle appliquée à la racine du document montre la création des éléments (problem, title, introduction, part) qui correspondent à la structure (voir figure 2) du document XML qui doit être produit.

```
<xsl:template match="/">
  <xsl:element name = "problem">
    <xsl:element name = "title">
      Enoncé
    </xsl:element>
    <xsl:element name = "introduction">
      <xsl:apply-templates select="//declared-construction-list"/>
      <xsl:apply-templates select="//figure"/>
    </xsl:element>
    <xsl:element name = "part">
      <xsl:apply-templates select="//questions"/>
      <xsl:apply-templates select="//solutions"/>
    </xsl:element>
  </xsl:element>
</xsl:template>
```

Figure 8 : création de la structure du document intermédiaire

La règle XSL, *<xsl:apply-templates select="//declared-construction-list"/>*, déclenche l'exécution des règles correspondant à chaque construction (point, parallélogramme, …). Dans la figure 9 on retrouve la création d'un élément *para* et le texte qui apparaîtra dans la présentation de l'activité. Pour écrire ce texte, il est nécessaire de recourir à des procédures puisque les mêmes processus, comme la recherche du nom d'un élément géométrique simple ou composé, sont généralement utilisés plusieurs fois. Pour écrire le nom d'un quadrilatère, il suffit d'appeler la procédure *quadrilateral-name* accompagnée de la référence contenue dans l'attribut *constructionid* de la construction correspondante.

```
<xsl:template match="quadrilateral[@type='parallelogram']">
<xsl:apply-templates />
  <xsl:element name = "para">
  Soit <xsl:call-template name='quadrilateral-name'>
          <xsl:with-param name="ref" select="../@constructionid" />
       </xsl:call-template> un parallélogramme
  </xsl:element>
</xsl:template>
```

Figure 9 : appel de la procédure d'écriture des noms d'éléments géométriques

Nous résumons dans la figure 10 les deux pas, transformation et présentation nécessaires au rendu dynamique et appliqué au cas particulier de notre exemple de géométrie.

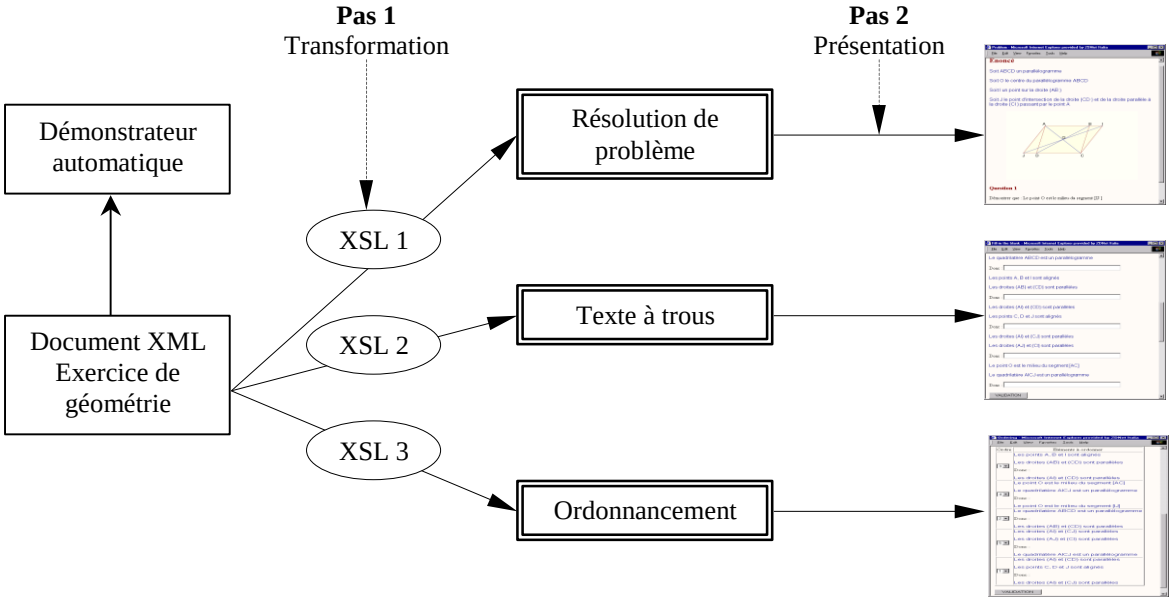

Figure 10 : les deux pas du rendu dynamique d'une activité

Nous y faisons également apparaître l'utilisation du document de base par un logicel. Ceci n'a pas été réalisé mais est tout à fait envisageable. En effet, DemAuto [5], logiciel d'aide à la démonstration de problèmes de géométrie conçu à l'IREM de Rouen et paru en 1993, contient un démonstrateur automatique. C'est un logiciel ouvert qui permet de construire les éléments d'un exercice grâce soit à des menus déroulants, soit au chargement de fichiers. La syntaxe de ces fichiers est spécifique, mais il suffirait de modifier la procédure de lecture pour adapter ce logiciel à notre type de document de base écrit en XML.

7. Conclusion

Un site destiné à l'apprentissage à distance doit permettre à des divers auteurs de créer des activités pédagogiques basées sur desdocuments pédagogiques variés. Ils profitent ainsi des services rendus par la plate-forme comme la présentation dynamique des documents ou la gestion de cours. Il est possible ou bien de reprendre des types de documents de base existants ou bien d'en créer de nouveaux. Dans ce dernier cas, pour les ajouter à la plate-forme, il suffit d'écrire les feuilles de style qui permettent de transformer les nouveaux documents de base en activités.
Nous pensons que la DTD que nous avons élaborée pour définir les types d'activités devrait permettre d'obtenir facilement un consensus même auprès des auteurs désireux d'écrire des nouveaux types de documents de base. Le passage par ces structures ne semble pas trop contraignant si l'on songe au bénéfice que représente l'utilisation des outils que nous développons pour présenter les activités.
Deux préoccupations doivent guider à la conception de nouveaux types de documents. Tout d'abord l'usage particulier des documents par rapport à leur domaine (ils peuvent être utilisés par des logiciels spécifiques, par exemple) et d'autre part, l'usage des documents par la plate-forme. Leur structure doit pouvoir supporter le filtrage qui les transformera en activités.
Il serait possible d'écrire une transformation d'un document de base qui produise directement le code HTML de la présentation sans passer par le niveau intermédiaire mais ceci entraînerait un surcroît de travail inutile. Il faudrait reprendre tous les types d'adaptation pris en compte dans les présentations de chaque activité et les adapter au nouveau document.
L'exemple implémenté montre bien les avantages obtenus en suivant le procédé en deux étapes, transformation et présentation : interopérabilité du document, réutilisabilité des documents et des ressources offerts par une plate-forme.

8. Références

[1] Bloom : *Bloom's Taxonomy* http://www.wested.org/tie/dlrn/blooms.html

[2] M. Cenati, L. Sommaruga : *Interaction and Dialogue in the NURAXI On-line Education Environment: an Example of Collaborative Learning with Jigsaw*, Roles of Communicative Interaction in Learning to Model in Mathematics and Science, C-LEMMAS 1999, Ajaccio, Corsica, 15-18 April 1999

[3] Cognitivo2 : http://www.fse.ulaval.ca/fac/dpt/cours/ten62630/Cognitivo2/pagemenuprinc.htm

[4] Dalgarno, B. *Choosing learner activities for specific learning outcomes: A tool for constructivist computer computer assisted learning design*. Edtech'98 Proceedings, 1998 http://wwwasu.murdoch.edu.au/aset/confs/edtech98/pubs/articles/abcd/dalgarno.html

[5] L. Jolivet, J.L Leperd, C. Moulin : *Demauto, Logiciel d'aide à la démonstration en géométrie*. Marque déposée de l'IREM de ROUEN, 1993.

[6] J.J. McConnell, *Activity List*, 1999. http://www-cs.canisius.edu/~mcconnel/activity_list.html

[7] MBTI® and Psychological Type. http://www.itd.net.au/aboutmbti.htm

[8] C. Moulin : *Typology of shared documents in a Web-based learning environment*, Proceedings of the AIED'99 workshop on "Ontologies for Intelligent Educational Systems", pp. 58-65, 1999

[9] S. St Laurent, E. Cerami : *Building XML Applications*, McGraw-Hill, 1999

[10] LotusXsl : http://www.alphaworks.ibm.com/aw.nsf/techmain/lotusxsl

[11] Extensible Markup Language (XML) 1.0 - W3C Recommendation http://www.w3.org/TR/1998/REC-xml-19980210 - 10 February 1998

[12] XSL Transformations (XSLT) Version 1.0 - W3C Recommendation http://www.w3.org/TR/xslt - 16 November 1999